

클러스터 기반의 VOD 서버 상에서 동적 버퍼 분할을 이용한 효율적인 부하분산 기법

권춘자⁰ 김영진 최황규

강원대학교 전기전자정보통신공학부

kwoncj@mail.kangwon.ac.kr, vicgt@chollian.net, hkchoi@kangwon.ac.kr

An Efficient Load Balancing Technique Based on the Dynamic Buffer Partitioning in Cluster Based VOD Servers

Chun-Ja Kwon⁰ Young-Jin Kim Hwang-Kyu Choi

Department of Electrical and Computer Engineering, Kangwon National University

요 약

클러스터 기반의 VOD 서버는 동시에 여러 사용자에게 실시간으로 고품질 서비스를 위해 정교한 부하 분산 기술과 버퍼관리 기술을 요구한다. 본 논문은 클러스터 기반 VOD 서버에서 동적 버퍼 분할을 이용한 새로운 부하분산 기법을 제안한다. 제안된 기법은 사용자 요청을 처리하는 서비스 노드간의 버퍼 성능과 디스크 접근 빈도를 고려하여 노드 부하를 고르게 분산한다. 또한 동적 버퍼 분할 기법은 동일한 연속 매체에 접근하려는 여러 사용자에게 평균 대기시간을 감소시킬 수 있도록 버퍼를 동적으로 분할한다. 각 서비스 노드에서 동적 버퍼 분할로 발생하는 버퍼 정보를 유지함으로써 각 서비스 노드의 버퍼 재사용을 극대화할 수 있다. 시뮬레이션을 통해 제안된 기법이 효율적인 버퍼 관리로 노드 부하 균등화, 사용자 평균 대기시간 감소, 병행 사용자 수 증가 등의 성능 향상을 얻었다.

1. 서 론

오늘날 컴퓨팅 기술, 압축 기술, 저장 장치, 그리고 네트워크 장치 등의 기술적 발달과 함께 멀티미디어 정보에 대한 요청도 크게 증가하고 있다[1].

멀티미디어 연속 매체는 텍스트 데이터뿐 아니라 비디오, 오디오, 정지 화상 등 다양한 데이터로 이루어져 있으므로 용량이 크고 실시간 검색이 가능해야 한다[2].

멀티미디어 VOD 서비스를 위한 시스템은 멀티미디어 서버, 통신망, 단말기 등의 구성요소로 이루어진다. 이중 멀티미디어 서버는 대용량 저장장치와 VOD 서비스 병행 처리가 가능한 실시간 구조를 필요로 한다. 따라서 고 가용성 및 확장성을 지원하기 위한 클러스터 VOD 서버 구조가 제안되었다. 비교적 저가의 고성능 PC 또는 워크스테이션들을 고속의 네트워크로 연결한 클러스터 VOD 서버는 확장성과 신뢰성이 높고 가격 대 성능 면에서 우수하므로 많은 연구가 이루어지고 있다[2].

본 논문은 클러스터 기반의 VOD 서버 환경에서 효율적인 버퍼관리 기법과 이를 활용한 부하분산 기법을 제안한다. 버퍼 관리 기법은 멀티미디어 연속 매체의 특성을 고려하여 각 노드의 버퍼 활용률을 높일 수 있는 동적 버퍼 분할 기법이다. 부하 분산 기법은 클러스터 서버의 노드간 버퍼 성능과 디스크 접근 빈도를 고려하여 전체 부하를 고르게 분산하는 기법이다. 제안된 기법을 이용함으로써, 버퍼의 활용도를 높일 수 있고, 병행 사용자들의 평균 대기시간을 줄여 클러스터의 성능을 향상시킬 수 있다.

본 논문의 구성은 먼저 2장에서 관련연구를 기술하고, 3장에서 제안된 부하 분산 기법과 동적 버퍼 분할 기법에 대해 설명한다. 4장에서 제안된 알고리즘의 성능분석

을 수행하며, 마지막으로 5장에서 결론을 맺는다.

2. 관련연구

멀티미디어 서버는 크게 4가지 분야로 서버 구성, 데이터 배치, 디스크 스케줄링, 버퍼 관리 분야이다. 이 중에서 클러스터 서버 구성의 최우선 목표는 각 서버 노드간의 부하를 균등화하는 것이다. 각 노드의 부하를 균등화하기 위해서 일반적으로 사용되는 방법은 각 노드의 접속 수에 의한 것이다. 그러나 VOD 서버의 경우 접속 수가 노드의 부하를 의미하지 않는다. 각 노드의 중요한 부하는 디스크 접근 빈도 수이며, 각 노드의 버퍼 관리 기법에 따라 디스크 접근 수를 줄일 수 있으므로 디스크 접근 빈도 수와 사용자 접속 수가 비례하지 않는다. 따라서 기존의 부하 분산 방식을 클러스터 VOD 서버에 적용할 경우 부하의 불균등을 초래할 수 있다.

멀티미디어 객체는 연속 접근 특성을 갖는다. 따라서 멀티미디어 서버를 위한 많은 버퍼 관리 기법들이 제안되었다. 가장 대표적인 버퍼 관리 기법으로 버퍼 선인출 기법과 버퍼 교체 기법이 있다[3].

버퍼 선인출 기법은 일정량의 데이터 블록을 버퍼에 미리 저장하여 서비스를 수행한다. 기존의 버퍼 선 인출 기법으로는 버퍼 비분할 기법, 정적 버퍼 분할 기법, 적응 버퍼 분할 기법 등이 있다. 그러나, 전통적인 버퍼 선인출 기법을 사용할 경우 버퍼 낭비 등 효율적인 버퍼 관리가 불가능하다.

버퍼 교체 기법은 버퍼가 필요할 때마다 각 기법의 알고리즘에 따라 선택된 버퍼를 교체하는 방법이다. 여기에는 FCFS, LRU, MRU 등의 교체 알고리즘이 있다[3].

그러나, 최적 버퍼를 교체하는 것이 불가능하다는 단점이 있다. 이외에 연속 매체를 위한 대표적인 버퍼 관리 알고리즘으로 BASIC/DISTANCE, Generalized Interval Caching 방법이 있다[4, 5].

3. 동적 버퍼 분할을 이용한 부하 분산 기법

3.1 클러스터 서버 구조

본 논문에서 사용하는 클러스터 서버의 구조는 기존의 two-tier 방식을 변형한 것으로 그림 1과 같다. 일반적인 two-tier 방식에 부하 분산기를 추가한 구조로 리눅스 가상 서버 구조와 유사하며, 부하 분산기를 통한 효율적인 부하 분산이 가능하다는 장점을 갖는다.

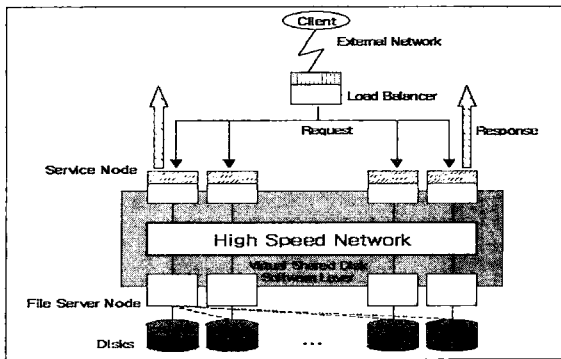


그림 1 클러스터 서버 구조

3.2 부하 분산기(Load Balancer)

부하 분산기의 목적은 클러스터 내의 서버간 부하를 분산시키는 것이다. 부하가 집중된 서버를 자동 감지해 다른 서버로 트래픽을 분산시킴으로 전체 서버의 성능을 균등화하고 빠른 응답속도를 유지하게 한다.

본 논문의 부하 분산기는 각 서비스 노드의 부하 정보와 열린 스트림 블록에 대한 정보, 동적 버퍼 정보를 이용해 서비스 노드간 부하를 적절히 유지하며 서버 노드의 성능을 최대한 높일 수 있도록 요청을 분배한다. 이 방법의 처리절차는 표1의 알고리즘과 같으며, 부하 분산 기법은 크게 2단계로 나누어 처리된다.

첫 번째, 사용자의 요청 파일을 서비스하고 있는 노드가 있는지 검색하여 해당 노드의 버퍼를 재 참조할 수 있다면 노드의 부하량에 상관없이 요청을 분배하는 단계이다.

두 번째, 요청을 재 참조할 수 있는 버퍼가 존재하지 않을 경우 각 서비스 노드의 부하량을 측정하여 부하가 가장 적은 노드로 요청을 분배하는 단계이다.

제안된 부하 분산 기법을 통한 요구 분배 과정은 그림 2와 같다. 부하 분산기는 각 서비스 노드의 버퍼 정보와 부하량 정보를 유지해야 하며 각 서비스 노드 정보는 표 2와 같다.

3.3 동적 버퍼 분할 기법

동적 버퍼 분할 기법은 연속 매체 스트림을 요청하는 다수의 사용자들에게 최소의 평균 대기시간을 제공할 수 있는 버퍼 선 인출 기법이다.

표 1 동적 버퍼 분할을 이용한 부하 분산 알고리즘

```

새로운 request 도착;
Enrollment Window Table를 검색;
If(request 처리 노드 존재 == TRUE)
    해당 노드로 요구 분배;
else
    {
        노드 관리 테이블 검색;
        if(가용 버퍼 존재 == TRUE)
            전체 그룹 수와 현재 그룹 수의 차이가 가장 큰 곳으로
            요구 분배;
        else if(최대 인터벌 > 0)
            해당 노드로 요구 분배;
        else
            대기큐에서 대기;
    }
    
```

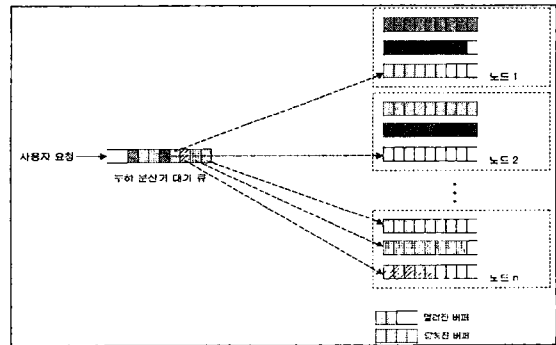


그림 2 부하 분산기의 요구 분배

표 2 부하 분산기에서 유지하는 각 노드의 정보

유지 정보	설명
Node ID	각 노드를 식별할 수 있는 식별자
각 노드의 그룹 수	각 서비스 노드에서 실질적으로 사용하고 있는 그룹 수
최대 그룹 수	각 노드에서 제공될 수 있는 최대 그룹 수
가용 버퍼량	각 노드에 남아있는 버퍼의 양
최대 버퍼 인터벌	각 노드의 서비스 그룹들 내에서 사용되는 버퍼의 최대 참조 간격
열려진 Enrollment Window	서비스 노드의 버퍼를 재 참조할 수 있을 경우 재 참조가 가능한 영화에 대한 이관과 상태 정보

부하 분산기를 통해 분배된 요구는 각 서비스 노드에 의해 처리되며, 각 서비스 노드는 동적 버퍼 분할 기법을 통해 사용자의 요구를 처리한다. 본 논문에서의 동적 버퍼 분할 기법은 처음에는 전체 버퍼를 하나의 버퍼로 관리하고, 필요할 때마다 버퍼를 그룹으로 분리하여 관리한다. 이 기법의 알고리즘은 표 3과 같다.

표 3 동적 버퍼 분할 알고리즘

```

if(새로운 request를 위한 버퍼 == free)
{
    새로운 request를 위한 버퍼할당;
    버퍼의 세크먼트들을 참조하여 진행한 마지막 사용자까지
    하나의 그룹이 됨;
    if(request == 각 그룹의 처음 request)
        저장 장치로부터 새로운 블럭을 읽으면서 진행;
    else
        첫 request를 따라 버퍼를 재 참조하며 진행;
}
else
{
    최대 인터벌을 갖는 두 사용자를 선택;
    요구가 속한 그룹을 두개의 그룹으로 분리;
    두 요구 사이의 버퍼를 회수, 새로운 요구에 할당;
}
    
```

4. 성능평가

본 논문에서 제안한 부하 분산 기법과 동적 버퍼 분할 기법의 성능 분석을 위해 시뮬레이션 하였다. 시뮬레이션에 적용된 파라미터는 표 4와 같다.

표 4 시뮬레이션 파라미터

파라미터	값	단위
전체버퍼크기	500(100..1000)	Mbytes
압축된 블럭의 크기	0.06	Mbytes
재생률	4	Bblocs/sec
디스크속도	20 (20,30,...100)	Mbytes/sec
영화 재생 시간	6000	sec
평균서비스요청간격	10(5,10,...30)	sec
Refresh Slack	0.012	block

각 그래프에서 DBP-LB는 동적 버퍼 분할 기법에 부하 분산 기법을 적용했음을 나타내고 DBP-RR은 동적 버퍼 분할 기법에 라운드 로빈 분배 방식을, GIC-RR을 Generalized Interval Caching 기법에 라운드 로빈 분배 방식을 적용했음을 나타낸다. 또한 GIC-LB는 Generalized Interval Caching 기법에 부하 분산 기법을 적용한 결과이다.

그림 3은 서비스 시간 경과에 따라 각 노드의 부하량(디스크 접근 빈도)이 어떻게 변하는지를 나타낸다. 그래프에서 보면 서비스 시간이 증가해도 그룹의 수가 일정하게 유지되는 것을 볼 수 있다.

그림 4는 노드 수를 5개로 유지하고 각 서비스 노드의 버퍼량을 증가시켰을 때 평균 대기시간을 측정한 그래프이다. 제안된 DBP-LB 기법이 DBP-RR 기법보다 성능이 우수함을 볼 수 있다. DBP-LB는 같은 영화에 대해 버퍼 재참조가 가능한 쪽으로 요구 분배가 이루어지므로 버퍼량이 커질수록 평균 대기시간이 줄어든다.

5. 결론

클러스터 VOD 서버에서 각 서비스 노드의 부하를 적절히 분산하기 위해서는 서비스 노드의 접근 빈도 수를 이용한 부하 분산 방법이 필요하다. 따라서 한번 읽은

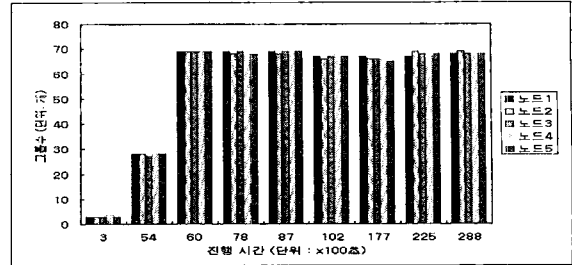


그림 3 서비스 진행 시간에 따른 각 노드의 부하량

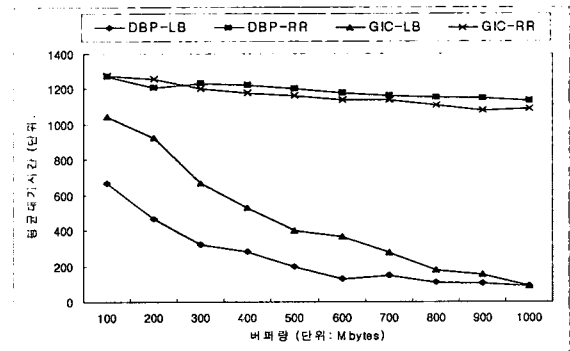


그림 4 버퍼량 변화에 따른 평균 대기시간 측정

연속 매체 데이터를 버퍼링하여 재 사용하면 저장 장치의 접근 부하를 줄일 수 있다.

본 논문은 시뮬레이션 성능 분석을 통하여 제안된 기법이 기존의 다른 기법에 비하여 부하량을 적절히 조절하면서, 평균 대기시간을 감소시킬 수 있음을 보였다.

참고 문헌

- [1] Wu D.P. Hou Y.W.T. and Zhu W.W., "Streaming video over the Internet: Approaches and directions", IEEE Transactions on Circuits & Systems for Video Technology, Vol. 11 No. 3, 2001.
- [2] 최재영, 최종명, "고가용성 리눅스", 한국 정보처리학회, Vol. 6, 1999.
- [3] R. Tewari, R. Mukherjee, "Design and Performance Tradeoffs in Clustered Video Servers", Proceedings of the International Conference on Multimedia Computing and Systems, Los Alamitos, CA, 1996.
- [4] Rotem and J. L. Zhao. "Buffer Management in Multimedia Database Systems", In Proc. of IEEE International Conference on Multimedia Computing and Systems, Hiroshima, Japan, June 1996.
- [5] A. Dan and D. Sitaram, "A generalized interval caching policy for mixed interactive and long video environments", In IS&T SPIE Multimedia Computing and Networking Conference, San Jose, CA, January 1996.